TECHNICKÁ UNIVERZITA

Fakulta mechatroniky a mezioborových inženýrských studií

DIPLOMOVÁ PRÁCE

Využití Internetu pro monitorování a řízení procesů

Usage of the Internet for monitoring and controling the processes

Liberec 2004 Petr Sidor

Anotace:

Cílem diplomové práce je zhodnotit možnosti Internetu pro monitorování a řízení procesů a zrealizovat ukázkové řešení.

Vlastní realizace řešení spočívala ve vytvoření WWW stránky, která pomocí UDP datagramů komunikuje s PLC Tecomat, který provádí vlastní řízení dynamického systému, kterým je model železnice.

Abstract:

The aim of the diploma thesis is to take into account the possibilities of the usage of Internet for monitoring and controlling the processes. Furthermore realization of an model solution.

The actual realization of the model solution is based on creating a website, which with the help of UDP datagrams communicates with PLC Tecomat. PLC Tecomat then itself executes the control of the dynamic system. The system is a model of a railway.

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum

Podpis

Seznam zkratek:

ADIR – Accessory Decoder information request
ADOR – Accessory Decoder operation request
ARPANET – Advanced Research Project Agency Network
bps – bit per sekund – přenosová rychlost v bitech za sekundu
CSS – Cascading Style Sheet
DNS – Domain Name System
HTML – Hyper-Text Markup Language
HTTP – Hyper-Text Transfer Protocol
IP – Internet Protocol
ISO – Mezinárodní normalizační úřad
LOR – Locomotive Operation Request
NCP – Network Control Protocol
NMRA – National Model Railroad Association
PHP – Personal Home Page
PLC – Programmable Logic Controller
RFC – Request For Comments
TCP – Transmision Control Protocol
TTL – Time to live
UDP – User Datagram Protocol
WWW – World Wide Web

Obsah:

1. Úvod

V současné době se pro řízení procesů využívá mnoho různých typů komunikačních systémů, jako jsou sériové linky RS – 232, RS – 485, nebo různé typy průmyslových sběrnic např. CAN, ProfiBus, ale zatím se příliš nevyužívá technologie největší světové sítě – Internetu.

Na tento stav reaguje diplomová práce, která má za úkol posoudit možnosti Internetu, zhodnotit jeho vhodnost pro řízení a na modelovém příkladě ukázat vlastní proces řízení.

Realizace ukázkového řízení bude provedena na PLC TC – 700 firmy TECO a.s.. Tento automat je vybaven síťovým rozhraním a podporuje komunikaci přes protokol UDP. Právě tohoto protokolu bude využito při vzájemné komunikaci mezi WWW stránkou a PLC. Za pomoci této stránky se bude uskutečňovat samotné řízení, nebo jen zadávání požadovaných hodnot, které bude mít PLC za úkol zpracovat a následně reagovat. Pro sledování aktuálního stavu železnice bude využita webová kamera, která bude součástí stránky.

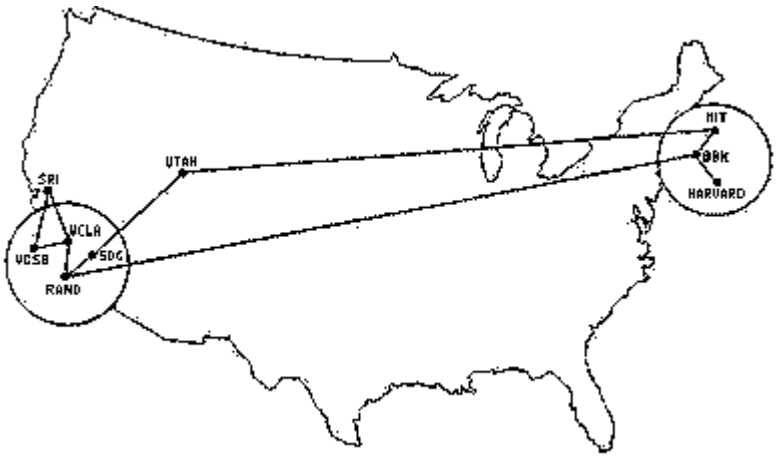
Pro vlastní tvorbu WWW stránky bude využito především skriptovacího jazyka PHP, který umožňuje posílání UDP datagramů. Aby bylo možné použít PHP, bude nutné nainstalovat a zprovoznit webový server a do něj začlenit samotný PHP. Samotná stránka pak bude, kromě jazyka HTML, tvořena ještě s využitím skriptovacího jazyka Javascript a s kaskádovými styly CSS.

2. Struktura Internetu

2.1. Historie Internetu

Počátkem 60. let se začali v USA objevovat myšlenky na vytvoření sítě, která by propojovala nejdůležitější vojenské, vládní a akademické počítače. Tato síť měla fungovat bez hlavního řídícího centra a měla být schopna provozu i v případě výpadku některého uzlu, např. vlivem jaderného úderu.

V Srpnu 1969 byla v USA, za finanční podpory Pentagonu, respektive jeho agentury DARPA (Defense Advanced Research Project Agency), uvedena do provozu siť ARPANET, která obsahovala 4 uzly, které byly na amerických univerzitách, a komunikovala rychlostí 500 bps. V roce 1971 se počet uzlů rozrostl na 15 a v roce 1972 již bylo propojeno celkem 37 počítačů. Většina komunikace spočívala ve výměně informací a proto byl vytvořen e-mail, aby došlo ke zjednodušení této komunikace.



Obr. 2.1 ARPANET v roce 1970

V roce 1973 se zapojili první dvě neamerické instituce a sice britská University College of London a norská Royal Radar Establishment. S narůstající velikostí ARPANETu bylo jasné, že současný síťový protokol NCP (Network Control Protocol) přestává stačit a vyhovovat rostoucím nárokům. Proto v roce 1974 byla zveřejněna první specifikace TCP (Transmision Control Protocol), která měla nahradit stávající protokol NCP. TCP byl robustní protokol, který měl dokonce zajišťovat opravu chyb při přenosech dat. Nakonec se ale TCP rozdělil na více protokolů, IP (Internet Protocol), který zajišťoval přenos dat, TCP, který využíval služeb IP a napravoval chyby při přenosu, a UDP (User Datagram Protocol), což byla varianta TCP, která ale nezajišťovala opravu chyb přenosu. V roce 1983 se oficiálně přešlo od NPC, výhradně k sadě protokolů TCP/IP.

Během rozvoje ARPANETu se k němu začali připojovat další sítě, které vznikali i mimo USA, a tak vznikala síť, které se začalo říkat ARPA Internet, později jen Internet a toto jméno vydrželo až do dnes.

V roce 1984 již bylo připojeno více než 1000 uzlů. Další velký průlom nastal v roce 1986, kdy na popud vládní agentury NFS (National Science Foundantion) vzniká NFSNET za silné podpory akademické obce. Tato nově vzniklá síť umožňovala přenosovou rychlost až 56 kbps. Díky podpoře NFS se do sítě zapojují další univerzity a výzkumná střediska a v roce 1989 je překročeno více než 100 000 uzlů. Také v Evropě vznikají první větší sítě a v roce 1983 je spuštěna EARN (European Academic and Research Network), která se brzo připojuje k americkému ARPANETu, respektive NFSNETu. Vznikají další sítě, např. EUNET (European UNIX Network), japonská síť JUNET a britská síť JANET (Joint Academic Network).

V roce 1989 vzniká protokol HTTP (Hyper-Text Transfer Protocol) a služba WWW (World Wide Web), která se stala nejrozšířenější službou Internetu. V tehdejší době však nebyla tak oblíbená jako dnes. Tato služba umožňovala vytvářet původně jen textové dokumenty a ty pak zpřístupnit pomocí serverů. Bylo

však nutností znát IP adresu serveru na kterém se daný dokument nachází. To bylo velice nepohodlné a vedlo to k zavedení DNS (Domain Name System), který umožňoval každé adrese přidělit jméno.

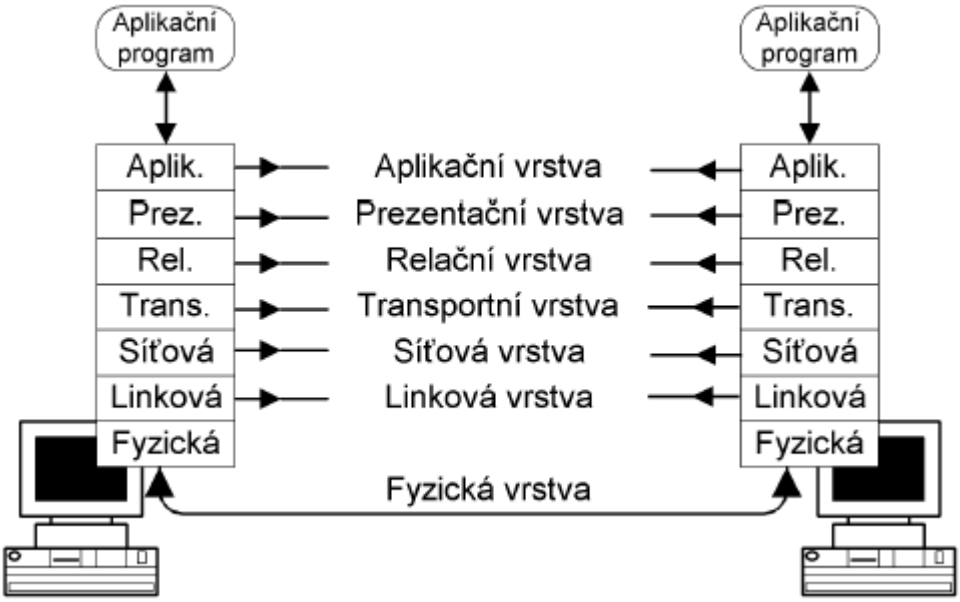
Následně dochází k rychlému rozšiřování, NFSNET dosahuje v roce 1991 kapacity 44,7 Mbps. V roce 1992 připojen více jak 1 milion uzlů. Celosvětové síti se začíná říkat Internet. Také dochází k připojení tehdejší ČSFR k EARNu a přidělení domény .cs a to v roce 1990. O něco později vzniká akademické síť CESNET. Také se začínají objevovat první komerční aktivity na Internetu. V roce 1993 dochází k zprivatizování Internetu (zpočátku pouze v USA), což mělo za následek zpřístupnění Internetu podnikům a hlavně veřejnosti. Počet připojených uzlů je více než 2 miliony, přes 600 webových stránek, objevuje se první grafický WWW prohlížeč – Mosaic. V roce 1996 již 10 mil. uzlů, 0,5 mil. webových serverů. Rok 2000 a 100 mil. uzlů, na 25 mil. WWW serverů, první linka 2,5 Gbps v ČR.

2.2. Síťové protokoly

Síťové protokoly jsou definované pomocí normy RFC – Request For Comments. Mezinárodní normalizační úřad (ISO) normalizoval soustavu protokolů označovaných jako ISO OSI model. Tento model se však v praxi nevyužívá, pouze slouží jako obecný model pro návrhy protokolů, které přišli později.

2.3. OSI model

OSI model se skládá ze sedmi vrstev. Tyto vrstvy jsou: fyzická, spojová, síťová, transportní, relační, prezenční, aplikační. Schématické znázornění komunikace mezi dvěma počítači je zobrazeno na obr. 2.2.



Obr. 2.2 Model OSI, převzato z [1]

Fyzická vrstva – zajišťuje vlastní fyzický přenos bitů, popisuje elektrické a optické signály použité při komunikaci mezi počítači. Definuje konektory, kabely, napětí, kódování signálů.

Spojová vrstva – někdy též označována jako linková. Řeší logiku práce s daty. Provádí paketizaci, detekci chyb a přístup k přenosovému médiu.

Síťová vrstva – řeší směrování v síti – hledání cest, vyvažování zátěže sítě, řídí práci sítě. Z hlediska síťové vrstvy jsou pakety adresovány adresou počítače (síťovým rozhraním).

Transportní vrstva – implementována v koncovém počítači – může přizpůsobit vlastnosti sítě (vrstvy 1-3) potřebám aplikace. Na této vrstvě dochází k rozlišení jednotlivých aplikací. Zpravidla bezchybný kanál zachovávající pořadí dat. Spravuje spojení. Adresace probíhá na úrovni jednotlivých aplikací, aplikace jsou jednoznačně adresovány v rámci jednoho počítače.

Relační vrstva – zabezpečuje výměnu dat mezi aplikacemi, přátelské ukončení spojení.

Prezenční vrstva – zabývá se významem přepravovaných dat, zodpovídá za reprezentaci a zabezpečení dat, kódování, komprimaci, šifrování.

Aplikační vrstva – definuje protokoly pro všeobecně používané služby, v jakém formátu a jak mají být data předávána/ přebírána od aplikačních programů, např. elektronická pošta, přenos souborů ...

2.4. Protokol IP

IP (Internet Protocol) je komunikační protokol, na kterém je dnes postaven Internet. Díky podpoře jednotného IP je umožněno kterémukoliv zařízení komunikovat se všemi ostatními. IP odpovídá síťové vrstvě a tak zajišťuje komunikaci dvou počítačů. Komunikace probíhá předáváním tak zvaných IP datagramů. Každý IP datagram obsahuje hlavičku a data, která přenáší. Součástí hlavičky je IP adresa, respektive dvě adresy, adresa příjemce a odesílatele datagramu. IP adresa je 4B číslo, které je celosvětově jednoznačné, což právě umožňuje jednoznačnou identifikaci i v tak rozsáhlé síti jakou je Internet. IP adresy se zapisují pomocí tečkovaného desítkového zápisu: 147.230.128.82. Protože však tento zápis je pro člověka nevhodný k zapamatování, je nahrazován textovým řetězcem, který je pak pomocí DNS serverů (Domain Name System) přeložen na IP adresu.

2.5. IP – datagram

IP – datagram se skládá ze záhlaví a přenášených dat. Záhlaví je velké 20B, pokud obsahuje volitelné položky, může být větší. Maximální velikost celého IP – datagramu je 65535 bytů.

0	8	16	24	32
Verze IP	Délka záhlaví	Typ služby	Celková délka IP – datagramu	
Identifikace IP – datagramu			Příznaky	Posunutí fragmentu – 13 bitů
TTL – 8 bitů		Protokol	Kontrolní součet z IP – záhlavý	
IP – adresa odesilatele (source IP – adress)				
IP – adresa příjemce (destination IP – adress)				
Volitelné položky záhlaví (nepovinné)				
Přenášená data (nepovinné)				

Tab. 2.1. Struktura IP – datagramu

Jednotlivé položky IP – datagramu:

- verze (*version*) – dlouhá 4 bity, obsahuje verzi IP, v současnosti je verze 4
- délka záhlaví (*header length*) – obsahuje délku záhlaví, neuvádí se však počet bajtů, ale jen čtvrtina, tzn. pokud je záhlaví velké 20B, pak délka záhlaví je 5. Maximální délka záhlaví pak může být maximálně 60B = 4bity *4 = 15 * 4. Protože povinné položky mají 20B, pak nepovinné mohou mít až 40B.
- typ služby (*Type of service – TOS*) – určuje požadavky na přepravu, např. šířku pásma, rychlost odezvy apod., ale v praxi nenašla uplatnění
- celková délka (*total length*) – obsahuje celkovou délku IP – datagramu v bytech. Maximální velikost je 65535 bytů.
- Identifikace IP – datagramu (*identification*) – společně s položkami *příznaky* a *posunutí fragmentu* využívají mechanismus fragmentace datagramu. Tento mechanizmus umožňuje rozdělit datagram na několik fragmentů, které mají stejnou identifikaci, ale liší se posunutím fragmentu. Jednotlivé fragmenty se také stanou datagramy a ty se pak nezávisle přepravovují. O složení jednotlivých fragmentů se pak stará příjemce datagramu.

- posunutí fragmentu (*fragment offset*) – určuje, na jaké pozici původního datagramu začínají data tohoto fragmentu.
- příznaky (*flags*) – 3bity | 0 | DF | MF |, DF (*Don‘t fragment*) – zakazuje fragmentaci, MF (*More fragment*) – nastaven na 1 u všech fragmentů kromě posledního.
- doba života datagramu (*Time to live – TTL*) – slouží k zamezení nekonečného přemisťování IP – datagramu. Každý směrovač, přes který datagram projde, sníží hodnotu TTL alespoň o 1. Pokud klesne na 0, pak se datagram zahodí a odesílateli IP – datagramu se pošle ICMP zpráva, že byl datagram zahozen kvůli vypršení životnosti.
- Protokol vyšší vrstvy (*protocol*) – udává číselnou identifikaci protokolu vyšší vrsty, který využívá IP – datagram ke svému transportu.
- Kontrolní součet z IP – záhlaví (*header checksum*) – obsahuje kontrolní součet pouze záhlaví, nikoli celého IP – datagramu. Pokud dojde ke změně záhlaví, např. ve směrovači změnou TTL, musí se kontrolní součet přepočítat.
- IP – adresa odesílatele a IP – adresa příjemce (*source and destination adress*) – obsahuje 4B IP – adresu odesílatele a příjemce IP – datagramu.

IP je protokol bez záruk, to znamená, že nezaručuje zda data v pořádku dorazí k příjemci. Bezpečnost přenosu má na starosti protokol vyšší vrstvy, například TCP.

2.6. Protokol TCP

Protokol TCP je protokol vyšší vrstvy než je IP a odpovídá transportní vrstvě podle OSI modelu. Jeho funkce je propojení na úrovni aplikací, na rozdíl od IP, který slouží ke spojení dvou počítačů, tzn. že IP vytváří spojení mezi dvěma počítači a TCP pak vytvoří spojení mezi aplikacemi běžícími na těchto počítačích. To znamená, že TCP využívá IP jako protokol síťové vrstvy a proto se uvádí zkratka TCP/IP.

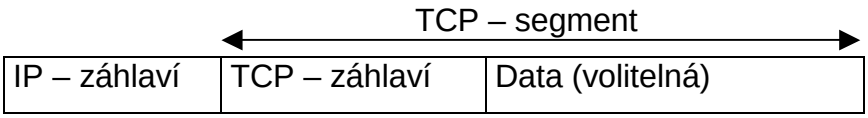
Protokol TCP je tzn. spojová služba, což znamená, že pokud chceme přenášet nějaká data je nejprve nutné vytvořit spojení – vytvoří se virtuální okruh. Toto spojení navazuje TCP klient, který se připojuje k TCP serveru. Vytvořený okruh je plně duplexní – možnost nezávisle přenášet data oběma směry.

TCP je protokol, který zaručuje spolehlivou přepravu. To je zajištěno mechanizmem potvrzování dat. V případě, že nedojde k potvrzení, budou data vyslána znovu. Zároveň je zde použit kontrolní součet, který zaručuje, že po přenosu nebudou data poškozena. Toto zabezpečení je proti poruchám technického rázu, nikoliv proti cíleným útokům.

Adresování TCP se provádí pomocí portů. Číslo portu je 16bitové číslo, proto mohou porty nabývat hodnot 0 až 65535. Celá adresa dat je tak složena z IP – adresy, která určuje, kterému počítači jsou data určena, a číslem portu, který určuje aplikaci na cílovém počítači. Jednoznačná identifikace spojení v Internetu v daném okamžiku je tak dáno zdrojovou, cílovou IP – adresou, zdrojovým, cílovým portem a protokolem.

2.6.1. TCP segment

Data se pomocí TCP přenášejí v TCP segmentech. Jelikož se TCP segment vkládá do IP – datagramu, může být jeho velikost maximálně 65535 bytů. Jednotlivé segmenty obsahují pouze část přenášených dat. O kterou část dat se v aktuálním segmentu jedná, určuje TCP záhlaví, respektive jeho položka *Pořadové číslo*. Na straně příjemce se segmenty skládají a potvrzuje se nejdelší souvislí prefix, který přijal od začátku vysílání. Pokud nedojde k doručení některého segmentu není nutné opět posílat všechna data, stačí pouze vyslat příslušný segment. Protože se při vysílání nečeká na potvrzení, neznamená případná ztráta potvrzení opakování dat, nýbrž při dalším potvrzení se potvrdí aktuální velikost přijatých dat.



0	8	16	24	32
Zdrojový port			Cílový port	
Pořadové číslo odesílaného bytu				
Potvrzení (pořadové číslo, číslo očekávaného bytu)				
Délka hl.		Příznaky	Délka okénka	
Kontrolní součet			Ukazatel naléhavých dat	
Volitelné položky				
Data				

Tab. 2.2. Struktura TCP segmentu

Jednotlivé položky TCP segmentu:

- Zdrojový port (*source port*) – port odesílatele TCP segmentu
- Cílový port (*destination port*) – port adresáta TCP segmentu.
- Pořadové číslo odesílaného bytu – udává pořadové číslo prvního bytu TCP segmentu v toku dat od odesílatele k příjemci. Pořadové číslo je dlouhé 4B, proto po dosažení hodnoty 2^{32}-1 začíná opět od 0. Při začátku vysílání dat se nezačíná od 0, ale od náhodně zvoleného čísla.
- Potvrzení – udává číslo očekávaného bytu, který má chce přijmout, tedy potvrzuje, že přijal vše do pořadového čísla bytu mínus jedna.
- Délka hlavičky – udává délku záhlaví v násobcích 4 bytu, podobně jako u IP
- Délka okénka – udává přírůstek pořadového čísla, který bude příjemce ještě akceptovat. Zvyšuje se tím efektivita – nemusí se čekat na potvrzení, zároveň se zabrání zahlcení pomalého příjemce. Odesílatel smí vysílat dokud nedosáhne velikosti okénka, poté musí čekat, dokud ji příjemce nezvýší. Pokud bude okénko prázdné je nutné zastavit vysílání a počkat na otevření.
- Ukazatel naléhavých dat – nastaven pouze pokud je nastaven příznak URG. Využívá se např. v protokolu TELNET. Používá se pokud chceme přednostně zpracovat některá data obsažená v TCP segmentu.

- Kontrolní součet – protože v IP – datagramu se počítá kontrolní součet jen z IP – záhlaví, je nutné pro zabezpečení integrity přenášených dat počítat kontrolní součet z celého TCP segmentu a některých položek IP - záhlaví, tzn. ze záhlaví a dat TCP segmentu a IP – pseudozáhlaví.

IP – adresa odesilatele (source IP – adress)		
IP – adresa příjemce (destination IP – adress)		
Binární nuly	Protokol	Celková délka IP – datagramu

Tab. 2.3. IP – pseudozáhlaví

- Příznaky – 6bitů

URG	ACK	PSH	RST	SYN	FIN

  - URG – segment obsahuje naléhavá data
  - ACK – segment má platné pole *Potvrzení*, nastaven vždy, kromě prvního segmentu, kterým se navazuje spojení.
  - PSH – signalizace, že segment nese aplikační data
  - RST – odmítnutí TCP spojení
  - SYN – odesílatel začíná s novou sekvencí číslování
  - FIN – odesílatel ukončil odesílání dat

2.6.2. Navazování a ukončování TCP spojení

Navazování spojení provádí strana, která se označuje jako klient. Druhá strana, která očekává spojení, se nazývá server a může se spojením souhlasit nebo ho zamítnout. Při navazování spojení musíme vždy znát číslo portu serveru, naopak port klienta určuje operační systém a může být pro jednu aplikaci při každém navazování spojení různý, ale během spojení se číslo portu nemění.

Vlastní navazování spojení probíhá tak, že klient vyšle první TCP segment na server. Tento segment je v celým spojení unikátní a to tím, že má nastaven příznak SYN a jako pořadové číslo odesílaného bytu je uvedena náhodně

vygenerovaná hodnota v rozmezí 0 až 2^{32}-1. Zároveň není nastaven příznak ACK, neboť nebyly žádná data na potvrzení. Druhým segmentem je segment odeslaný serverem, který již potvrzuje přijatá data (nastaven příznak ACK), což je ovšem jen pořadové číslo klienta. Zároveň obsahuje příznak SYN a vlastní vygenerované pořadové číslo odesílaného bytu. Třetí segment opět vysílá klient a v něm také potvrzuje pořadové číslo, tentokrát serveru. Tímto třetím segmentem končí navazování spojení, které bývá někdy označováno jako „třífázový handshaking“.

Ukončování spojení může provádět jak klient tak i server. Ukončování se provádí tak, že jedna strana (mohou i obě) vyšle segment s nastaveným příznakem FIN, který oznamuje, že ukončující strana již nebude odesílat žádná data. Ukončující strana se tak dostane do stavu který se označuje jako *aktivní ukončení spojení* (active close) Během tohoto stavu nemůže odesílat žádná data, ale stále bude potvrzovat přijímání dat. Druhá strana však stále může vysílat data, dokud sama neprovede ukončení => je ve stavu *pasivního ukončování* (pasive close). Dobu mezi aktivním ukončením a ukončením spojení nazýváme *polouzavřeným spojením* (half close). Pro řádné ukončení jsou potřeba čtyři segmenty, neboť dochází k potvrzování příznaku FIN podobně jako u navazování spojení. Tzn., že strana která ukončuje spojení vyšle segment s příznakem FIN, druhá strana odešle potvrzení, poté odvysílá vlastní data, která první strana potvrzuje. Nakonec vyšle druhá strana segment s příznakem FIN, první strana potvrdí a tím dojde k řádnému ukončení.

2.6.3. Shrnutí TCP

Výhody TCP:

- snadná detekce nedoručení dat
- garantována správnost pořadí přijímaných dat.
- nemůže vzniknout duplicita dat.
- zajištěná správnost dat. (Společně s daty je odesílán také kontrolní součet.)

Nevýhody TCP:

- příliš mnoho řídících informací - hlavička TCP obsahuje mnoho informací (kontrolní součet, pořadí, a další informace nutné pro přenos v rámci spojení).
- velká zátěž pro síť – TCP záhlaví je poměrně dost velká, nutný přenos dat opačným směrem (potvrzování). Každé potvrzení je další TCP paket poslán opačným směrem.

<table>
<tr><td>Verze IP</td><td>Délka záhlaví</td><td>Typ služby</td><td colspan="2">Celková délka IP – datagramu</td></tr>
<tr><td colspan="3">Identifikace IP – datagramu</td><td>Příznaky</td><td>Posunutí fragmentu – 13 bitů</td></tr>
<tr><td colspan="2">TTL – 8 bitů</td><td>Protokol</td><td colspan="2">Kontrolní součet z IP – záhlavý</td></tr>
<tr><td colspan="5">IP – adresa odesilatele (source IP – adress)</td></tr>
<tr><td colspan="5">IP – adresa příjemce (destination IP – adress)</td></tr>
<tr><td colspan="5">Volitelné položky IP – záhlaví (nepovinné)</td></tr>
<tr><td colspan="3">Zdrojový port</td><td colspan="2">Cílový port</td></tr>
<tr><td colspan="5">Pořadové číslo odesílaného bytu</td></tr>
<tr><td colspan="5">Potvrzení (pořadové číslo, číslo očekávaného bytu)</td></tr>
<tr><td>Délka hl.</td><td></td><td>Příznaky</td><td colspan="2">Délka okénka</td></tr>
<tr><td colspan="3">Kontrolní součet</td><td colspan="2">Ukazatel naléhavých dat</td></tr>
<tr><td colspan="5">Volitelné položky</td></tr>
<tr><td colspan="5">Data</td></tr>
</table>

Tab. 2.4. Ukázka celé datové struktury TCP/IP protokolu

Velmi používanou alternativou k TCP je protokol UDP.

2.7. Protokol UDP

Protokol UDP je stejně jako TCP protokolem transportní vrstvy a stejně využívá IP jako síťový protokol. Stejně jako TCP je určen ke komunikaci mezi dvěma aplikacemi, které se identifikují pomocí čísla portu, který je na jednom počítači vždy jedinečný. Avšak může být na jednom počítači stejné číslo portu pro TCP a UDP, což je dáno tím, že čísla TCP a UDP portů jsou na sobě nezávislá.

Hlavní rozdíl oproti TCP je ten, že UDP je nespojová služba, tedy nenavazuje spojení. Odesílatel odešle UDP datagram příjemci a už se nestará, zda byl doručen, ani nepřijde žádné potvrzení. Také zde není žádná kontrola poškození dat, neboť kontrolní součet se počítá pouze z UDP záhlaví a IP pseudozáhlaví. Dokonce je možné kontrolní součet úplně vypnout. Také zde nedochází k číslování segmentů jako u TCP, takže může dojít k tomu, že data přijdou ve špatném pořadí, nebo může dokonce dojít k duplikaci dat. Odstranění těchto chyb je pak věcí samotné aplikace.

Proč se tedy používá UDP, když je tak nespolehlivý? Jeho výhoda je především v malém záhlaví a paradoxně v tom, že nedochází k potvrzování. Následkem toho dochází při přenosu k malému zatížení sítě, což v některých aplikacím může být výhodné. Například při přenosech zvuku v reálném čase (přehrávání internetového rádia) není nutné potvrzování dat, pokud by došlo ke ztrátě UDP datagramu, tak pouze může zvuk přeskočit, zaprskat apod., ale nic závažného se nestane. Avšak je dosaženo veliké snížení zátěže sítě. Další možností UDP je možnost odesílat data na několik skupin stanic najednou. UDP se také používá při komunikaci s DNS servery, právě z důvodu nízké zátěže pro server.

2.7.1. UDP datagram

Záhlaví UDP je velice jednoduché. Obsahuje pouze čísla zdrojového a cílového portu, délku dat celého UDP datagramu a kontrolní součet. Celé UDP záhlaví je tak 8 bytů.

<table>
<tr><td>Verze IP</td><td>Délka záhlaví</td><td>Typ služby</td><td colspan="2">Celková délka IP – datagramu</td></tr>
<tr><td colspan="3">Identifikace IP – datagramu</td><td>Příznaky</td><td>Posunutí fragmentu – 13 bitů</td></tr>
<tr><td colspan="2">TTL – 8 bitů</td><td>Protokol</td><td colspan="2">Kontrolní součet z IP – záhlavý</td></tr>
<tr><td colspan="5">IP – adresa odesilatele (source IP – adress)</td></tr>
<tr><td colspan="5">IP – adresa příjemce (destination IP – adress)</td></tr>
<tr><td colspan="5">Volitelné položky IP – záhlaví (nepovinné)</td></tr>
<tr><td colspan="3">Zdrojový port</td><td colspan="2">Cílový port</td></tr>
<tr><td colspan="3">Délka dat</td><td colspan="2">Kontrolní součet</td></tr>
<tr><td colspan="5">Data</td></tr>
</table>

Tab. 2.5. Ukázka celé datové struktury UDP/IP protokolu

2.8. Vhodnost Internetu pro řízení

Výhoda řízení přes Internet spočívá hlavně v tom, že při dnešní veliké dostupnosti Internetu je možnost řídit daný systém v podstatě odkudkoliv a to buď přes klasický web, kdy stačí zadat adresu s příslušnou stránkou, nebo pomocí řídící aplikace. Vytvoření takovéto řídící aplikace je dnes relativně snadné, neboť většina programovacích jazyků již obsahuje nástroje pro naprogramování přenosu dat přes Internet a to buď protokolem TCP nebo UDP. Stejně tak je možné vytvořit i webovou stránku, která umožňuje přenos dat určeným směrem a s požadovanými daty, tedy ne jen běžnou komunikaci mezi stránkou a serverem.

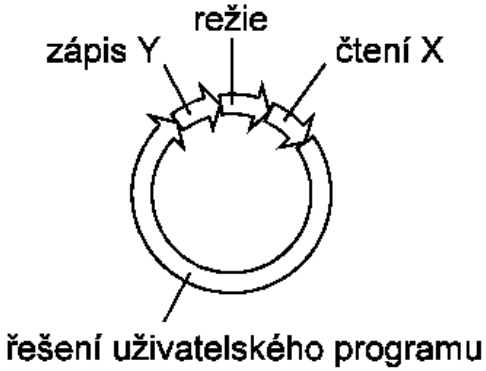
Při řízení přes WWW stránku je však těžší naprogramovat vizualizaci. Je to způsobeno jednak tím, že se hůře reaguje na události, jednak tím, že část zdrojového kódu se zpracovává na straně serveru a část na straně klienta.

Využitelnost Internetu při řízení spočívá spíše v možnosti dálkového řízení, tedy spíše nastavování, zadávání požadovaných hodnot, nežli podílením se přímo na samotném procesu řízení. To je způsobeno především tím, že i když je zaručen relativně spolehlivý přenos dat, již není zaručena doba za kterou data dorazí, proto je nevhodné, aby se Internetu využívalo pro přímé řízení nebo regulaci.

3. Programovatelný automat Tecomat

Programovatelné automaty (PLC) Tecomat firmy TECO a.s. jsou číslicové řídící systémy vhodné pro řízení technologií v různých oblastech průmyslu. Automaty mohou být vybaveny řadou vstupů, výstupů a to jak analogových tak i digitálních (reléových).

Vlastní řízení pak probíhá díky cyklicky vykonávanému uživatelskému programu, který je uložen v paměti automatu. Cyklus začíná načtením hodnot ze vstupních modulů PLC do zápisníkové paměti X. Poté se provede uživatelský program. Následně se na výstupní moduly zapíší hodnoty výstupních proměnných. Nakonec je čas pro režii, kdy se provádí příprava na řešení dalšího cyklu. Celý jeden cyklus je znázorněn na obr. 3.1.



Obr. 3.1

3.1. Základní charakteristiky TC – 700

PLC TC – 700 je modulární programovatelný automat, tzn. že je složen z několika typů modulů, které tvoří celý automat. Toto řešení umožňuje sestavit si PLC přímo na míru pro svoje požadavky. Základ PLC sestává z nosného rámu, napájecího modulu a centrální jednotky. Dále je možné dovybavit PLC systémovými komunikačními moduly, analogovými vstupními a výstupními moduly, binárními vstupními a výstupními moduly. Těchto modulů může být zapojeno veliké množství.

Centrální jednotka obsahuje:

- obvod reálného času
- paměť uživatelského programu a tabulek – 64 + 64 kB
- záložní paměť programu EEPROM
- přídavná paměť dat DATABOX – 128 kB, rozšířitelná až na 3 MB
- uživatelské registry – 40 kB
- zálohování RAM a RTC – 20 000h
- časovače – rozsah 65536 x 10 ms ÷ 10 s s možností kaskádování
- čítače – rozsah 65536 s možností kaskádování
- podpora čítačů a časovačů IEC
- sériové kanály
  – kanál CH1 – režim PC – komunikace s nadřízeným systémem prot. EPSNET
  – režim PLC – sdílení dat mezi PLC v síti EPSNET-F
  – režim uni – obecný kanál s libovolnou komunikací
  – režim MPC – výměna dat s podřízenými PLC v síti EPSNET multimaster
  – režim MDB – komunikace s nadřízeným systémem prot. MODBUS
  – kanál CH2 – režim EIO – připojení dalších čtyř periferních rámů
  – režimy PC, PLC, uni, MPC, MDB
- rozhraní USB – režim PC
- rozhraní Ethernet 10 Mb – režim PC

3.2. Síť EPSNET

V síti EPSNET mohou pracovat dva druhy stanic: nadřízená stanice (*master*) – aktivní účastník, který řídí komunikaci a podřízená stanice (*slave*) – pasivní účastník, jenž odpovídá na dotazy nadřízené stanice. Vzájemná komunikace pak probíhá na principu dotaz – odpověď, kdy master vyšle dotaz a slave mu pošle odpověď.

Síť EPSNET umožňuje dvě základní konfigurace: *monomaster* – v síti je jen jedna nadřízená stanice a několik podřízených maximálně 126, *multimaster* – v síti se nachází několik nadřízených stanic a několik podřízených. Nadřízenou stanicí bývá většinou počítač s vizualizačními nebo vývojářskými aplikacemi, nebo systém Tecomat / Tecoreg se sériovým kanálem v režimu MAS nebo v režimu MPC. Podřízenými stanicemi jsou pak systémy Tecomat / Tecoreg se sériovým kanálem v režimu PC. Při nastavování vlastností komunikačního kanálu se kromě režimu nastavuje také rychlost přenosu, adresa pro komunikaci, doba prodlevy odpovědi, detekce CTS a parita.

3.2.1. Obecná struktura protokolu EPSNET

Komunikace ve směru nadřízená stanice → podřízená stanice (zpráva, dotaz):

a) zpráva bez datového pole

SD1	DA	SA	FC	FCS	ED

b) zpráva s datovým tokem

SD2	LE	LER	SD2R	DA	SA	FC	DATA	FCS	ED

Komunikace ve směru podřízená stanice → nadřízená stanice (potvrzení, odpověď):

a) odpověď bez datového pole

– krátké pozitivní potvrzení

SACK

pevná hodnota $E5

– jiný typ potvrzení

SD1	DA	SA	FC	FCS	ED

b) odpověď s datovým polem

SD2	LE	LER	SD2R	DA	SA	FC	DATA	FCS	ED

SD1 – úvodní znak 1 – pevná hodnota $10

SD2 – úvodní znak 2 – pevná hodnota $68

LE – délka dat – počet bytů položek DA+SA+FC+DATA, tj. 3 … 249

LER – opakovaná délka dat – stejná hodnota jako LE

SD2R – opakovaný úvodní znak 2 – pevná hodnota $68

DA – cílová adresa – hodnota pro nad(pod)řízenou stanici 0 … 126

SA – zdrojová adresa

FC – řídící byte rámce

DATA – vlastní datové tělo zprávy – maximálně 246 bytů

FCS – kontrolní součet – bytový součet všech bytů položek DA, SA, FC a DATA se zanedbáním vyšších řádů vzniklých přenosem

pro SD1 $\sum_{DA}^{FC} \bmod 256$ pro SD2 $\sum_{DA}^{FCS-1} \bmod 256$

ED – koncový znak – pevná hodnota $16

Ukázka navázání komunikace se stanicí s nastavenou adresou 0. Nadřízené stanici určíme adresu např. 126. Ukázka vychází ze struktury protokolu bez datového pole:

zpráva $10 $00 $7E $69 $E7 $16

odpověď $10 $7E $00 $00 $7E $16

Význam jednotlivých bytů:

1. byte - úvodní znak – pevná hodnota 10h
2. byte – cílová adresa – zpráva pro stanici s adresou 0
   (odpověď pro stanici s adresou 126 = 7Eh).
3. byte – zdrojová adresa – adresa nadřízené stanice
   (v odpovědi adresa podřízené stanice)
4. byte – řídící byte rámce – pro navázání spojení → 69h,
   pro odpověď → 0h
5. byte – kontrolní součet
6. byte – koncový znak – pevná hodnota 16h.

3.2.2. Protokol EPSNET přes UDP

Při komunikaci s PLC přes ethernetové rozhraní se využívá komunikační protokol UDP s číslem portu 61682. V jednom okamžiku je umožněna komunikace až čtyřem účastníkům současně, ale jen jeden může využívat systémové služby, jako je např. programování PLC. Vlastní komunikace začíná navázáním spojení a je ukončena po 5 sekundách od skončení přenosu dat.

Data v UDP datagramu zasílaném do PLC začínají záhlavím, které má velikost 6B, a dále pokračuje zpráva ve standardním protokolu EPSNET. Zprávy jsou stejné jako ty, kterými se komunikuje přes sériovou linku. V jednom UDP datagramu lze za jedním záhlavím poslat až 5 zpráv EPSNETu a tím více zhustit komunikaci. Odpověď na dotaz má stejný formát. Nejprve záhlaví a za ním následuje příslušný počet odpovědí v protokolu EPSNET. Data v UDP datagramu musí mít sudou délku, proto pokud má zpráva EPSNET délku lichou je nutné na konec dat v UDP datagramu doplnit 0.

Záhlaví:

1., 2.B – číslo relace – vysílá Master a PLC kopíruje zpět do odpovědi.
Může sloužit pro stanici Master pro případnou kontrolu

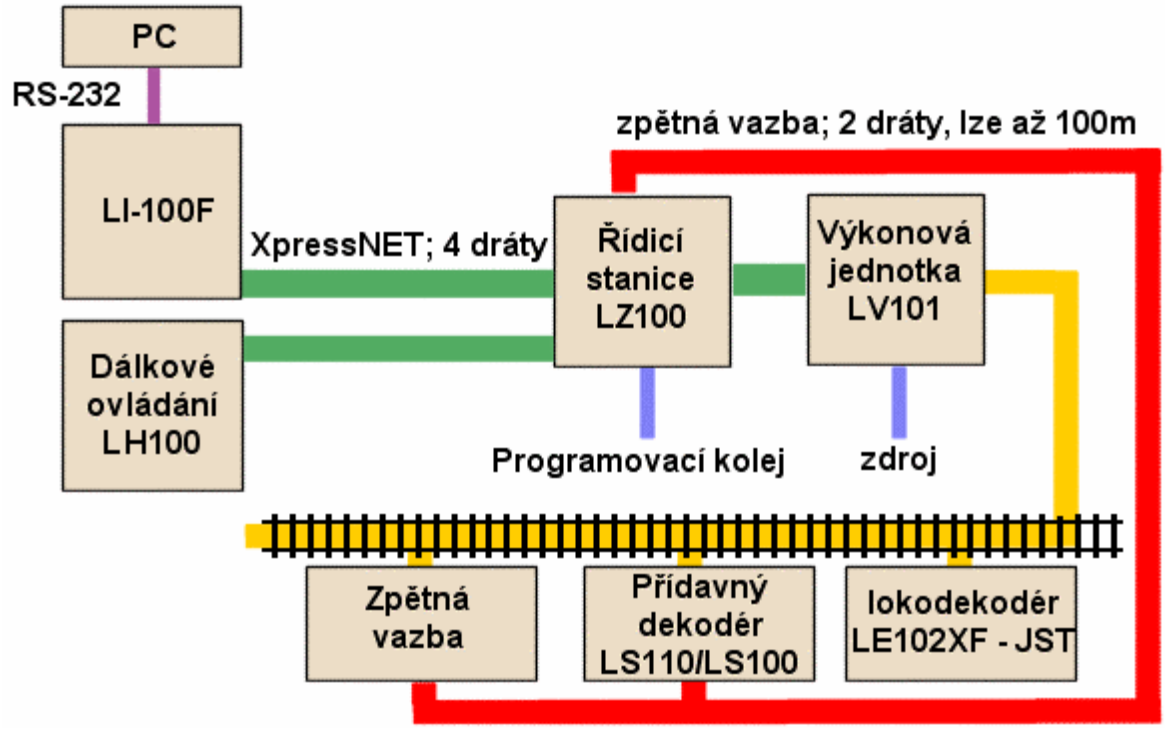
3. B – číslo protokolu. Pro režim PC (EPSNET slave) = 2.

4. B – nevyužit

5., 6. B – počet bytů následujících dat.

4. Modelová železnice

Modelová železnice na Technické Univerzitě v Liberci v učebně TK3 je složena z kolejišť, výhybek, lokomotiv, zvedáků od firmy Fleischman a celý model železnice je ovládán pomocí řídícího systému firmy Lenz.

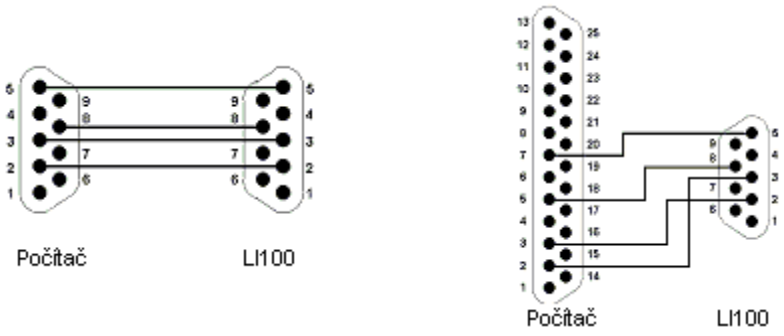


Obr. 4.1 Blokové schéma modelové železnice, převzato z [4]

Hlavní jednotkou je řídící stanice LZ100, na kterou jsou připojeny vstupní zařízení LI-100F, které zprostředkovává komunikaci s PC, a dálkové ovládání LH100. Jako výstupní jednotky slouží přídavné dekodéry, které ovládají výhybky, zvedáky apod., a lokodekodéry, sloužící pro řízení mašinek. Řídící stanice ovládá všechna připojená zařízení prostřednictvím XpressNet, kdy podle požadavku vstupních zařízení vygeneruje řídící signál. Tento signál je vyslán přes výkonovou jednotku LV101, kde je signál zesílen a zároveň namodulován na napájecí napětí, do kolejnic. Díky tomu jsou přídavné dekodéry a lokodekodéry z kolejnic nejen napájeny, ale i řízeny.

4.1. Modul LI100F

Modul LI100F je jednotka sloužící k zprostředkování komunikace mezi PC a řídící stanicí. Obsahuje rozhraní pro komunikaci přes RS-232 a zároveň pro komunikační protokol XpressNet, který je založen na RS-485. Jednotku je možné připojit k počítači přes devíti nebo dvacetipěti pinový konektor, viz obr 4.2.



Obr 4.2. Propojení LI100F s PC přes 9 a 25 pinový konektor

Vlastní komunikace mezi PC a LI100F probíhá s přenosovou rychlostí 9600 Bd, nebo 19,2 kBd, pro řízení komunikace je využit CTS hardwarový handshake a datový rámec je složen z 1 start bitu, 8 datových bitů, 1 stop bitu a paritního bitu, který je však při přenosu nevyužit.

	START BIT (log 0)	0	1	2	3	4	5	6	7	PARITY BIT	STOP BIT (log 1)	
		DATA BITS										

Tab 4.1. Datový rámec

4.2. Modul LZ100

Modul LZ100 je hlavní jednotkou celého řídícího systému. Zpracovává informace ze vstupních zařízení (LH100, LI100,…) a na základě požadavků těchto zařízení generuje řídící signály. Tyto signály jsou vyslány do výkonové jednotky LV101, kde se řídící signál zkombinuje s napájecím napětím, aby dosáhl požadovaného výkonu a následně je vyslán do kolejiště, na které jsou připojeny přídavné dekodéry (LS100/LS110) a dekodéry pro lokomotivy (LE102XF).

4.3. Modul LS100/ LS110

Moduly LS100 a LS110 jsou použity jako přídavné dekodéry, kde LS100 je bez zpětné vazby na rozdíl od LS110, který zpětnou vazbu zabudovanou má. Na dekodéry je možno připojit až 4 výhybky/ zvedáky plus 4 zpětné vazby u LS110. Jednotlivé stavy výhybek a zvedáků jsou ovládány přímo přídavným dekodérem, a to tak, že řídící stanice vyšle požadavek na přídavný dekodér (ADOR – Accessory Decoder operation request). Požadavky na dekodér budou popsány v kap. 4.7.2. Modul také může zpětně poskytnout informace o stavu výhybek a zvedáků, pokud dostane od řídící stanice dotaz na přídavný dekodér (ADIR – Accessory Decoder information request), viz kapitola 4.7.3.

4.4. Modul LE102XF – JST

Modul LE102XF – JST slouží jako univerzální lokodekodér pro ovládání mašinek. Umožňuje vybírat mezi 14/27, 28/55 a 128 krokovým rychlostním módem. Je vybaven 2 funkčními výstupy, které mohou být zatíženy až 200mA. Lokodekodér podporuje rozšířenou adresaci a pokročilé skládání řízení, které se ovládá pomocí příkazů na lokodekodér (LOR – locomotive operation request), viz kapitola 4.7.1.

V lokodekodéru jsou uloženy informace nutné pro řízení mašinek (např. adresa, maximální rychlost, rozjezdové křivky, apod.), které je možné programovat pomocí dálkového ovladače LH100, nebo přímo z PC.

4.5. Lokomotiva

Každá lokomotiva je vybavena lokomotivním dekodérem LE102XF – JST, který je přes kola a kolejnice napojen na řídící jednotku, respektive na výkonovou jednotku, z které přijímá řídící signály. Tyto signály ovládají střídavý motor v lokomotivě a přídavná zařízení (světla, rozpojky). Pro identifikaci lokomotivy, resp. lokomotivního dekodéru, je dle standartu NMRA možno použít krátkou adresaci (pro identifikaci lokomotivy 1 – 127) nebo dlouhou adresaci (0 – 16383). Pokud by byly použity obě metody adresace, docházelo by k překrývání, což by způsobovalo vážné komplikace při komunikaci. Proto došlo k úpravě jednotlivých adresací, kdy krátká adresace má rozsah 1 až 99 a dlouhá adresace 100 až 9999. Model železnice, který je k dispozici, disponuje pouze krátkou adresací.

4.6. Výhybky a zvedáky

Výhybky a zvedáky se k řídícímu systému připojují přes přídavné dekodéry LS100/LS110. Pro přehození výhybky je nutné použít dva řídící pakety. Při obdržení prvního paketu, pustí přídavný dekodér proud do cívky výhybky, čímž dojde k přitažení relé a přehození výhybky. Poté musí následovat (do 2s) druhý paket, který vypne přívod proudu do cívky. Pokud by k tomu nedošlo, tak by se výhybka zničila. Stejným způsobem se ovládají zvedáky a závory, s tím rozdílem, že zde může být delší časový interval mezi jednotlivými pakety.

4.7. Komunikace s modelovou železnicí

Komunikace s modelovou železnicí probíhá prostřednictvím XpressNet posíláním 4B nebo 6B datových paketů. První byt je vždy hlavička, která identifikuje celý paket a zároveň informuje o jeho délce. Např. pokud bude první byt B4h, znamená to, že se paket týká lokomotivy a bude obsahovat 4 datové byty. Nakonec paketu se přidává kontrolní součet XOR, který slouží jako jednoduché zabezpečení. Kontrolní součet se provádí z celé zprávy, tedy i z hlavičky. Celá zpráva je tedy dlouhá 6B.

4.7.1. LOR – Příkaz na lokodekodér

Datový paket LOR je dlouhý 6B, 1.B je hlavička B4h, 4B jsou datové, 6.B je kontrolní součet XOR.

hlavička	adresa lokomotivy	1.datový B	2.datový B	Rychlostní mód	XOR
B4h	0 – 64h	ovládání	0	14, 27, 28	xor

Tab 4.2. Datový paket LOR

Bit					Rychlost	Bit					Rychlost
4	3	2	1	0		4	3	2	1	0	
0	0	0	0	0	0	0	1	0	0	1	8
0	0	0	1	0	1	0	1	0	1	0	9
0	0	0	1	1	2	0	1	0	1	1	10
0	0	1	0	0	3	0	1	1	0	0	11
0	0	1	0	1	4	0	1	1	0	1	12
0	0	1	1	0	5	0	1	1	1	0	13
0	0	1	1	1	6	0	1	1	1	1	14
0	1	0	0	0	7						

Tab 4.3. Tabulka rychlostí

1. datový B:

0	Směr (1b)	Světla (1b)	Rychlost (5bitů)

2. datový B: nevyužit, může sloužit pro ovládání dalších funkcí lokomotivy

Rychlostní mód: možnost zvolit jeden ze tří rychlostních módů, v našem případě výhradně využíváme 14-ti rychlostní mód ~ 0h.

4.7.2. ADOR – Příkaz na přídavný dekodér

Datový paket pro ADOR je dlouhý 4B, kde 1.B je opět hlavička s hodnotou 52h, následují dva datové byty a nakonec opět XOR. V prvním datovém bytu je adresa přídavného zařízení, s nímž chceme komunikovat, vydělená čtyřmi. Tímto dostaneme adresu dekodéru, na který je zařízení připojeno. Konkrétní adresa zařízení je zbytek po dělení, takže možné adresy jsou 0, 1, 2 a 3. Toto číslo se ještě násobí dvěma, abychom dostali hodnotu o jeden bit posunutou doleva. Takto upravená adresa zařízení je pak uvedena ve 2.datovém bytu. Navíc jsou v tomto bytu uvedeny informace, co se má s daným zařízením provést.

Hlavička	1.datový B	2.datový B	XOR
52h	adresa dekodéru	definování požadavku	xor

Tab. 4.4. Datový paket ADOR

2. datový B:

bit 7 – vždy nastaven na 1 **1**0001xx1b

bit 3 – určuje stav zapojení

– u výhybek zapnutí/ vypnutí proudu 1000**1**xx1b / 1000**1**xx1b

bit 2,1 – určuje výstupní přídavné zařízení 10001**xx**1b

bit 0 – určuje propustný směr červený/ zelený 10001xx**1**b

4.7.3. ADIR – dotaz na přídavný dekodér

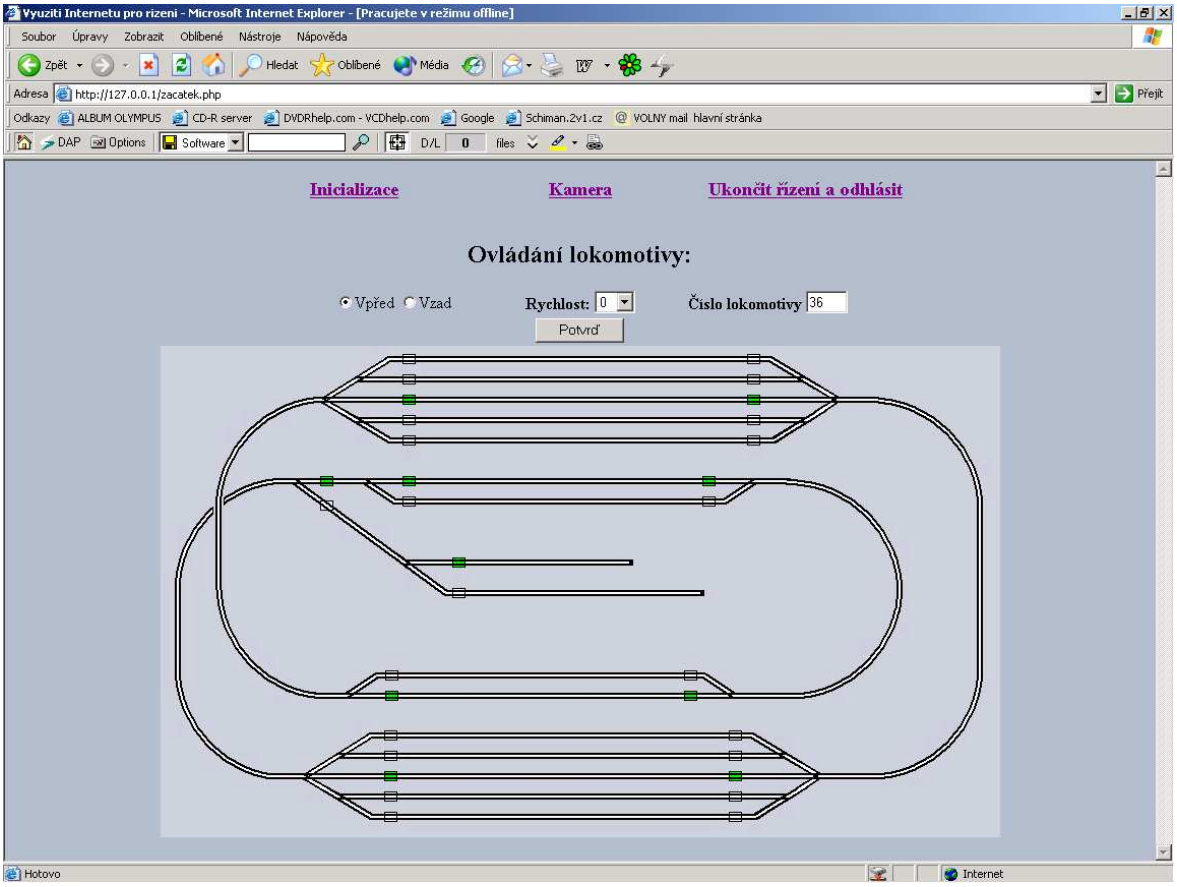
Při obdržení této 4B dlouhé zprávy vyšle přídavný dekodér zpět na řídící jednotku odpověď o svým stavu ve stejném formátu jako je ADOR. Následně řídící jednotka tuto odpověď poskytne dále. Následné rozpoznání a zpracování je už na uživateli.

Hlavička	1.datový B	2.datový B	XOR
42h	adresa dekodéru	80h+N	xor

Tab. 4.5. Datový paket ADIR

5. Tvorba webové stránky

Pro ověření možnosti řízení pomocí Internetu, byla vytvořena webová stránka, na které je zobrazeno schéma modelu železnice. Toto schéma umožňuje ovládání jednotlivých prvků kolejiště. Zároveň je na stránce zobrazen formulář pro řízení libovolného počtu lokomotiv. Také je přidána možnost zapnout webovou kameru, která zobrazuje v reálném čase aktuální stav modelu.



Obr. 5.1 WWW stránka pro řízení modelu železnice.

5.1. Instalace webového serveru

Aby bylo možné vytvořené stránky nejen zobrazit na jednom počítači, ale také je zpřístupnit přes Internet, je nutné na počítači, kde budou dané stránky k dispozici, nainstalovat některý z tzv. webových serverů, tedy programů, který zaručí právě onu dostupnost stránek přes Internet. Na výběr je hned několik programů, nejznámější jsou např. Apache, který je volně šiřitelný, nebo IIS (Internetová Informační Služba), která je součástí některých operačních systémů Windows (např. Windows XP Professional).

Jelikož vytvořené stránky budou komunikovat přes server s PLC, pracovat se soubory, je nutné nainstalovat některý skriptovací jazyk, který toto umožňuje. Pro svou snadnost a široké možnosti se velmi využívá skriptovací jazyk PHP, který je také zdarma. PHP však není možné použít samostatně, nýbrž se vždy instaluje právě jako součást webového serveru.

5.1.1. Instalace Apache HTTP Server

Instalační program pro server Apache je možné stáhnout na adrese http://www.apache.org v sekci HTTP Server, kde stáhneme verzi pro Win32. Samotnou instalací nás provede sám program. Po nainstalování je nutné provést některé úpravy v konfiguračním souboru *httpd.conf*, který nalezneme v adresáři, kam byl Apache nainstalován (implicitně c:\Program Files\Apache Group\Apache). Zde je nutno vyplnit položku *DocumentRoot*, kam zadáme umístění souborů pro webové stránky. Soubor httpd.conf uložíme a spustíme Apache. Pokud proběhlo vše v pořádku, zobrazí se ve webovém prohlížeči, po zadání adresy http://127.0.0.1, obsah adresáře, který jsme nastavili jako místo pro stránky.

5.1.2. Instalace PHP

PHP skriptovací jazyk je možné stáhnout na adrese http://www.php.net v sekci download, kde vybereme opět verzi pro win32. Stažený archív rozbalíme do nějakého adresáře např. c:\PHP\. Poté je nutné zkopírovat soubor php.ini-dist pod jménem php.ini do adresáře, kam je nainstalován operační systém např. c:\windows\. Následně v tomto souboru doplníme informace o umístění stránek na disku a to tak, že za položku doc_root zadáme stejný adresář jako v případě instalace Apache. Jako poslední krok je nutné následující řádky vložit do konfiguračního souboru Apache httpd.conf.

```
ScriptAlias /php/ "c:/php/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"
```

Pro ověření správnosti instalace vytvoříme zkušební soubor, který nazveme např. *phpinfo.php* a do něj zapíšeme tyto rádky:

```
<?php
        phpinfo();
?>
```

Pokud vše funguje správně, měly by se, po zadání internetové adresy http://127.0.0.1/phpinfo.php do webového prohlížeče, zobrazit informace o nastavení PHP. Tímto celá instalace skončila, pokud bychom chtěli rozšířit PHP o některé další moduly, nebo se při instalaci vyskytli nějaké problémy, tak podrobný návod na instalaci najdeme v souboru *install.txt* v adresáři c:\php\.

5.2. Úvod do HTML

Základ většiny webových stránek je tvořen pomocí jazyka HTML (Hyper-Text Markup Language), který umožňuje jednoduchým způsobem vytvářet webové stránky s obrázky, odkazy, tabulkami, formuláři a dalšími prvky. Neumožňuje však dynamicky reagovat na podněty od uživatele, proto se často do stránek vkládají skripty, které právě umožňují dynamické chování stránek. Skriptovacích jazyků je celá řada, nejznámější jsou Javascript, VBScript, které pracují na klientském počítači, nebo PHP, Pearl, které běží na serveru. Postupem času se také začalo využívat tzv. kaskádních stylů CSS (Cascading Style Sheet), které umožňují rozsáhlé změny formátování stránek.

Obecná struktura HTML dokumentu má tvar:

<html>
<head>
definuje hlavičku stránky
sem patří: název stránky, skripty, styly, definování znakové sady, apod.
</head>

<body>
uvozuje tělo stránky
sem patří: publikovaný text, obrázky, tabulky, odkazy… znova skripty
</body>
</html>

Část <head></head>

– pojmenování stránky: <title>název stránky</title>

– nastavení stylů na stránce: <style>definice stylu</style>

– jednoduchá funkce v JavaScriptu na otevření stránky v novém okně:

```
<script>
    function otevri(stranka)
    { window.open(stranka); }
</script>
```

Část <body></body>

– základní formátování:

 nový řádek <hr> vodorovná čára

text kurzívou text tučně

<u>podtržený text</u> odstavec

<h1>hlavička</h1> centrovaný text

– tabulky:

<table>tabulka</table>

<tr>definuje řádek tabulky</tr>

<td>definice buňky</td>

<th>hlavička</th>

– obrázky: <img src="název obrázku">

formát obrázku určují další parametry:

align – zarovnání vůči textu

width – šířka obrázku

height – výška obrázku

alt – textová náhražka

border – rám okolo obrázku

– odkazy: text odkazu

– formuláře: <form>celý formulář</form>

Navíc je nutno do tagu <form> vložit informace o tom, jakým způsobem budou data na server odeslána a kam budeme formulář odesílat. Metoda odeslání *post* nepřidá data z formuláře do internetové adresy, na rozdíl od metody *get*, která data začlení do adresy. Celá definice formuláře může vypadat takto:

<form action="cíl.php" method="post">

vstupní objekty formuláře:

<input type="text" name="text1"> - vstupní textové pole

<input type="submit" value="Odešli"> - tlačítko pro odeslání

<input type="reset" value="Smaž">

- tlačítko pro vymazání hodnot z formuláře

<input type="checkbox" checked name="souhlas">

- zaškrtávací políčko, implicitně zaškrtnuto

<input type="radio" name="r1" value="a">
<input type="radio" name="r1" value="b">

- dvě zaškrtávací políčka provázaná přes jméno, je odesláno buď *a* nebo *b*

<select name="roleta">
<option value="val1"> Položka 1
<option value="val2"> Položka 2
</select>

- rolovací seznam, kde vybíráme z předdefinovaných hodnot, ale je možno zadat i vlastní data. Pokud neuvedeme parametr *value*, bude odeslána přímo vybraná položka, respektive její název.

Příklad: Formulář pro ovládání pohybu lokomotivy.

V tomto příkladě je ve formě tabulky udělán formulář, kde jsou použity dvě zaškrtávací políčka pro směr lokomotivy, rolovací seznam pro zadání požadované rychlosti vlaku, textové pole, kam se zadává číslo lokomotivy a nakonec tlačítko pro odeslání dat na server, kde je požadavek zpracován. Také je zde nástin formátování textu, kdy je použit formát hlavičky pro nadpis a zvýraznění textu pomocí tučného písma.

Kód takovéhoto formuláře vypadá následovně:

```
<h2>Ovladani lokomotivy:</h2>
<table align="center" width="50%" >
<form action="lokomotiva.php" target="poslat" method="post">
<tr><td align="center" width="33%">
<input checked type="radio" name="smer" value="0">Vpred
<input type="radio" name="smer" value="1">Vzad</td>
<td align="center" width="33%">
<b>Rychlost:</b>
<select name="Rychlost">
<option>0</option>
<option>1</option>
<option>2</option>
            :
<option>13</option>
<option>14</option>
</select></td>
<td align="center" width="33%">
<b>Cislo lokomotivy </b><input type="text" name="loko" size="3" value="36">
</tr>
<tr><td align="center" colspan=3>
<b><input type="submit" value="   Potvrd   "></b></td></tr>
</form></table>
```

5.3. Základy PHP

PHP patří do skupiny skriptovacích jazyků, které běží na straně serveru a je na něm závislé, neboť na něm běží jeho interpret. To je rozdíl např. od JavaScriptu, který se stahuje přímo s HTML stránkou a vykonává se prohlížečem na straně klienta. U PHP klient nemá přístup ke zdrojovému kódu, jako to je u HTML, nýbrž pouze k jeho výsledkům. Nevýhodou je, že PHP nedokáže dynamicky reagovat na události způsobené klientem (např. pohyb myši), pouze reaguje na požadavky odeslané klientem na server. Proto je nejvýhodnější kombinovat PHP s JavaScriptem.

Samotné PHP skripty se zapisují, stejně jako u JavaScriptu, přímo do HTML kódu, pouze je nutné označit kde začíná a končí PHP script. To se provádí značkami <?php a ?>, nebo je možné použít značku <script language="php">, následuje program a nakonec se vše uzavře značkou </script>. V PHP se velmi často procuje s proměnnými, které začínají znakem *$* (např. $jmeno). Výhodou je, že se proměnné nemusejí dopředu deklarovat, jako je tomu u vyšších programovacích jazyků. Ukončení jednotlivých příkazů se provádí pomocí *;*.

5.3.1. Základní příkazy PHP

Zde bude uveden pouze základní popis příkazů, použitých při tvorbě této diplomové práce. Kompletní seznam všech příkazů je možné najít na webové stránce www.php.net.

echo "text"; - výpis textu, je možné použít klasické HTML formátování, vypisování proměnných, např. echo "Součet 4 + 5 = " .(4+5);

for ($i;$i<10;$i++) - příkaz pro cyklus, který 10x vykoná *příkazy*.
{ příkazy }

if ($i<10) { *příkazy1* } - příkaz podmínky, pokud je splněna jsou vykonány
else { *příkazy2* } *příkazy1*, v opačném případě *příkazy2*.

$soubor=fopen("data.txt","r"); - otevření souboru pro čtení, pokud místo r bude w, pak se soubor otevře pro zápis.

fclose($soubor); - zavře soubor

fwrite($soubor,"data") – zapsání dat do souboru
fread($soubor,$počet); - přečte $počet bytů ze souboru

time(); - zjistí aktuální čas.

$fp = fsockopen("udp://adresa", $port, $errno, $errstr);

- otevře UDP spojení s počítačem se zadanou adresou a portem.

5.4. Omezení přístupu

Při řízení přes Internet vystává problém s vícenásobným přístupem a následnými konflikty v řízení. Proto bylo nutné zavést určitá omezení, která tento problém odstraní. Řešení spočívalo v umístění přihlašovacího dialogu na úvodní stránku, který kontroluje jednak oprávněnost přístupu, tzn. omezit přístup nepovolaným lidem, a dále pak kontroluje, zda již někdo jiný není přihlášen a neřídí systém.

S tímto však vyvstal další problém, který spočíval v tom, že může nastat okamžik, kdy již nikdo není připojen, avšak zároveň nedošlo k odhlášení, což mohlo být způsobeno, buď zavřením webové stránky, nebo jakýmkoli přerušením spojení. Proto je nutné kontrolovat zda je ještě někdo připojen, což ovšem není z principu webu možné. Řešení spočívá v tom, že každá akce na stránce vyvolá událost, která si ukládá informace o době akce. Pokud nenastane nějaká akce po určitou dobu, dojde po dotazu k odhlášení a odpojení uživatele. Tím je zaručeno, že nemůže dojít k zablokování stránky.

Přihlašovací dialog je tvořen klasickým formulářem, kdy se na server odesílá vyplněné heslo. To je vyhodnoceno společně se zjištěním, zda již někdo není připojen. Následně dojde buď k připojení, nebo oznámení, že není připojení možné.

Celou kontrolu je možné provést následujícím kódem:

```
<?php
```

```
$heslo = $_POST["heslo"];              - načtení přijatých dat a uložení do proměnný
$spravne=true;
if (file_exists("prihlasen.txt"))      - zjištění zda soubor existuje,
  { $fp=fopen("prihlasen.txt","r");    - otevření souboru pro čtení
    $cassouboru=fread($fp,filesize("prihlasen.txt"));    - načte celý soubor do proměnný
    $rozdil=time()-$cassouboru;        - rozdíl aktuálního času a času v souboru
    if ($rozdil < 600)                 - při rozdílu menším než 600s, připojení není možné
      { $spravne=false;}
  }
if ($heslo=="a" AND $spravne==true)        - ověří správnost hesla
    {    //kdyz je spravny heslo:
     $fp=fopen("prihlasen.txt","w+");  - otevření souboru pro zápis
     fwrite($fp,time());               - uložení aktuálního času do souboru
     fclose($fp);                      - uzavření souboru
     ?><script>location='inic.php';</script><?php     - načtení nové stránky
    }
    else { //kdyz je spatny heslo:
            if ($spravne==true)
              {echo " <b> Heslo je chybné! </B> "; }
              else {echo "<b> Nyni je nekdo pripojen, zkuste to pozdeji.<b>"; }
             ?><BR>                    - nový řádek
              <A href="index.html">Zpet</A><?php    - odkaz na přihlašovací stránku
         }
?>
```

5.5. Posílání dat

Při ovládání kolejiště, komunikuje WWW stránka prostřednictvím serveru s PLC. Tato komunikace probíhá přes protokol UDP, kde data mají strukturu komunikačního protokolu EPSNET.

Při odesílání řídících dat bylo možné použít dvě varianty řešení. První spočívala v kompletním odeslání všech řídících dat do PLC, který je následně

pošle do kolejiště. Druhá metoda spočívala v odeslání pouze požadavku, který by měl PLC zpracovat a podle něj vygenerovat příslušná řídící data. Pro řešení byla nakonec zvolena první varianta pro svoji jednodušší implementaci, neboť není nutné programovat PLC, a také z důvodu, že druhá varianta již byla řešena pouze s tím rozdílem, že vysílání požadavku bylo řešeno přes sériovou linku.

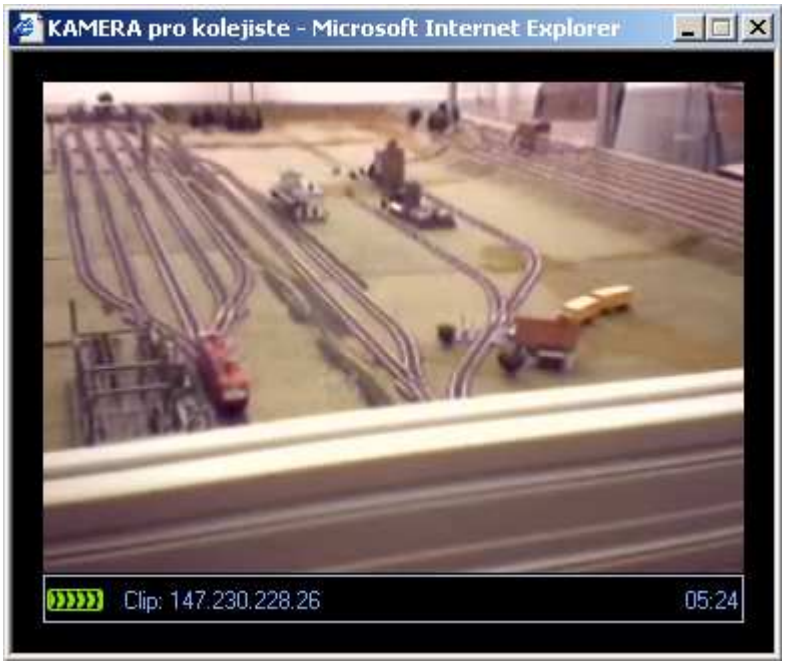
Vlastní řešení posílání dat pomocí PHP vypadalo následovně:

```
<?php
$datagram1=pole2string(vytvor_pole(0x52,5,0x8A,0xDD));  - vytvoření datového pole
$datagram2=pole2string(vytvor_pole(0x52,5,0x82,0xd5));
posliudp($datagram1,$datagram2);            - volání funkce pro odeslání dat

function posliudp($data1,$data2)
{ $fp = fsockopen("udp://147.230.128.82", 61682, $errno, $errstr,1);   - vytvoření spojení
  if (!$fp) {echo "ERROR: $errno - $errstr<br>\n";}
  else { fwrite($fp,$data1);                - odeslání dat pro přehoz výhybky
         sleep(1);                          - časová prodleva
         fwrite($fp,$data2);                - vypnutí proudu výhybkou
         fclose($fp);                       - ukončení spojení
}}

function vytvor_pole($a,$b,$c,$d)     - funkce pro vytvoření datového pole se strukturou
{ $crc=0;                                 protokolu EPSNET
$pole= array (0xab,0x02,0x02,0,0,34,0x10,0,0x7e,0x69,0xe7,0x16,0x68,22, 22,0x68,0,
                0x7e,0x63,0x0c,3,20,0,4,$a,$b,$c,$d,3,18,0,1,4,3,16,0,1,0xa0);
for ($i=16;$i<sizeof($pole);$i++)       - kontrolní součet
   {$crc=$crc + $pole[$i];}
$pole[sizeof($pole)]=$crc % 256;
$pole[sizeof($pole)]=0x16;
return($pole);
}

function pole2string($pole)    - funkce pro převod datového pole na řetězec znaků (bytů)
{ $retezec='';
```

```
 for ($i=0;$i<sizeof($pole);$i++)
  { $retezec=$retezec . chr($pole[$i]);}
 return($retezec);
}
?>
```

5.6. Webová kamera

Pro možnost sledovat aktuální situaci na kolejišti je použita webová kamera. Ta je připojena k serveru, na kterém je spuštěn program Windows Media Encoder. Tento program umožňuje zpracovávat obraz z kamery a dále ho poskytovat přes Internet. Umožňuje výběr z několika video formátů, nastavení velikosti obrazu, počet snímků za sekundu a celkový bitrate. Také se zde nastavuje velikost výstupního bufferu, který je nutno nastavit na co nejmenší možnou hodnotu, aby se dosáhlo co nejmenšího časového rozdílu mezi zobrazeným a odeslaným okamžikem.



Obr. 5.2. Webová kamera

Realizace zobrazení obrazu z webové kamery pomocí HTML:

```
<object id="MediaPlayer" width="300" height="350"
```

```
        classid:="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
codebase="http://activex.microsoft.com/activex/controls/mplayer/en/nsmp2inf.cab#Version=
            6,4,7,1112"
standby="Loading Microsoft Windows Media Player components..."
type="application/x-oleobject">
    <param name="AutoStart" value="True" />
    <param name="Filename" value="http://147.230.228.26:1028" />
    <param name="ShowControls" value="1" />
    <param name="ShowStatusBar" value="1" />
    <param name="ShowDisplay" value="1" />
    <param name="TransparentAtStart" value="1" />
    <param name="AutoSize" value="0" />
    <param name="DisplaySize" value="0" />
    <embed src="http://147.230.228.26:8080" width=350 height=270
      type="application/x-mplayer2"  name=MediaPlayer  autostart=1  showcontrols=0
      showstatusbar=1 autorewind=1 showdisplay=0>
    </embed>
</object>
```

6. Závěr

Pomocí vytvořené WWW stránky je možné relativně dobře daný systém řídit. Pouze není možné zadávat povely rychle za sebou. Je to způsobeno tím, že každý povel znamená načtení nové stránky, tedy jedno nové připojení na PLC a není možné vytvořit více než čtyři spojení. Samotné spojení trvá zhruba 5 sekund. Toto omezení je dáno samotným PLC.

Další nevýhodou je, že jednotlivé povely trvají dlouho. To je způsobeno tím, že není možné posílat povely rychle za sebou, neboť je pak řídící systém kolejiště nestačí zpracovat. PHP však neumožňuje dělat mezi povely menší časové intervaly než je jedna sekunda. Naopak tento problém částečně kompenzuje problém s omezením současného počtu spojení.

Navržené řešení řízení by bylo lepší pro řízení nějakého systému, kde by se pouze nastavovaly, zadávaly požadované hodnoty a po určitých intervalech se zobrazovaly hodnoty naměřené. Naopak pro řízení modelu železnice by byla vhodnější spíše aplikace vytvořená přímo pro danou úlohu.

Další možností by bylo tyto dvě řešení zkombinovat, tedy pomocí WWW stránky by se odesílaly na server pouze požadavky jednotlivých úkonů. Na serveru by pak byla spuštěna aplikace, která by přijímala požadavky a na jejich základě vysílala potřebná data do PLC. Jelikož by všechna data byla vysílána z jediné aplikace, nebyl by problém s omezením počtu připojení. Také by bylo možné zkrátit jednotlivé časy mezi povely. Toto řešení by mohlo byt vhodným námětem pro navázání a rozšíření této diplomové práce.

Literatura:

[1] Dostálek, L., Kabelová, A..: *Velký průvodce protokoly TCP-IP a systémem DNS*. Praha: Computer Press 2002. 435. ISBN 80-7226-323-4

[2] Broža, P.:*Programování WWW stránek pro úplné začátečníky*. Praha: Computer Press 2000. 172. ISBN 80-7226-278-5

[3] Bakken, S., S., Aulbach, A..: *PHP manuál*, 2001

[4] Válek, M.:*Prediktivní řízení modelové železnice – Diplomová práce*, Liberec: TU Liberec, 2003. 56.

Internetové adresy

[5] Firemní materiály firmy Lenz Elektronik, GmbH. http://www.lenz.com

[6] Firemní materiály firmy Teco, a.s. http://www.tecomat.cz

[7] PHP. http://www.php.net

[8] Jak psát web. http://www.jakpsatweb.cz